SECURITY REPORT FOR R5, THE RCO-POW FOR NEBULA

JOSEPH VAN NAME

Abstract. The cryptocurrency Nebula will use R5 as its proof-of-work problem. We shall briefly investigate the security and other aspects of the problem R5.

1. Introduction and general comments

In this report, we will analyze the security, efficiency, reversible device friendliness, and other characteristics of the proof-of-work problem R5 for the cryptocurrency Nebula.

We do not have the luxury to launch Nebula after giving a thorough cryptanalysis of R5 as was given for cryptosystems such as the AES or SHA-3 since we want to launch Nebula as quickly as possible and because we do not have the resources to perform such a thorough cryptanalysis of R5. It typically takes a few years of thorough analysis from the cryptography community from when a new symmetric cryptosystem is proposed until when the cryptosystem is commonly used in practice (public key cryptosystems typically a much take longer time to become ready for public use from when such a public key cryptosystem is proposed). Fortunately, due to the nature of the POW problem scheme R5, the security requirements for R5 are quite lax compared to the security requirements for other symmetric cryptosystems. Furthermore, the security of R5 is reinforced by a more than sufficient number of rounds in each of the problems R5 in order to compensate for a lack of cryptanalysis before launch (later in this report, we shall see that a larger number of rounds for R5 has other advantages besides providing security reinforcement).

The upcoming cryptocurrency Nebula is exceptional since an existing cryptosystem would not be sufficient as a POW problem for Nebula, and we would have used an existing cryptosystem if it would have been sufficient for Nebula. There are dangers in producing a new cryptosystem and implementing that new cryptosystem right after it is developed, and some developers have even foolishly implemented new cryptosystems into their cryptocurrencies. For example, the cryptocurrency IOTA which had a 3 Billion USD market cap on August 17, 2017 has in the past implemented an insecure hash function named Curl which has allowed adversaries to forge new digital signatures without access to the private key required to make those digital signatures. The hash function Curl was designed by the IOTA developers and has been susceptible to differential attacks.

The security requirements for a cryptocurrency POW are lower than they are for other symmetric cryptosystems such as cryptographic hash functions or symmetric encryption-decryption cryptosystems. After all, the POW scheme R5 for Nebula has to be solved within two minutes while cryptographic hash functions and symmetric encryption systems both need to remain secure indefinitely. Furthermore,

an adversary against a symmetric encryption-decryption cryptosystem may have an unlimited amount of ciphertexts which may aid in breaking the cryptosystem.

The miners will input the data $\mathbf{k}\#\mathbf{x}$ into some function f_i. However, miners will not have any control of the 256 bit hash $\mathbf{k}$ (other than trying many different hashes in order to select a suitable hash) and miners will only have control of the 68 bit string $\mathbf{x}$. Since miners have little control of the input $\mathbf{k}\#\mathbf{x}$ of f_i, it is unlikely that an adversary will be able to solve R5 more quickly than it will take simply by computing $f_i(\mathbf{k}\#\mathbf{x})$ as many times as possible.

An increased number of rounds in the functions for R5 provides benefits for RCO-POW problems which are not related to security; if a RCO-POW problem has more rounds, then a greater portion of the computing power will be spent on reversible processes as opposed to the irreversible process of resetting the data for every attempt of solving the RCO-POW problem. On the other hand, having an excessive number of rounds for Nebula is not ideal either since reversible computation may initially be excessively error prone and if there are too many rounds in any of the problems in R5, then it may be infeasible to compute $f_1, \ldots, f_5$ reversibly without making one of these errors. We shall therefore be fairly liberal with the number of rounds we put into each of the five POW problem R5 even if it were possible to acheive cryptographic security using a much smaller number of rounds, but we will not give any of the R5 problems an excessive number of rounds.

It is often the case that a cryptocurrency POW problem is slow to verify because the POW problem was selected in order to achieve some other purpose such as ASIC resistance or shorter block times (with shorter block times, one has to verify the solution to the POW problem more often so it will take a greater portion of computation power to verify all the solutions). For example, the cryptocurrency Dash uses X11 as its POW and X11 goes through 11 different hash functions in order to resist mining from specialized hardware; on the other hand, only one hash function is required for cryptocurrencies. The POW problem R5 was designed not necessarily to be the most efficient cryptocurrency RCO-POW problem to verify but instead the problem R5 was designed to incentivize the construction of the reversible computer as much as possible without compromising the security or efficiency of Nebula. Nevertheless, each of the functions $f_1, \ldots, f_5$ in R5 has a gate count under 100,000 so under this measure, R5 could be considered to be an efficiently verified POW problem.

We chose several problems in order to strategically guide the advancement of reversible computing hardware up to commercial use. The difficulty of constructing hardware that can solve Problem i will increase as i increases. In other words, it will likely be easiest for a hardware manufacturer to construct a reversible device for solving Problem 1 and it be much more difficult for a hardware manufacturer to construct a reversible device for solving Problem 5.

If our RCO-POW simply used a reversible cellular automaton as a randomizing function, then it is feasible that manufacturers would construct reversible computers which are good at simulating 2D reversible cellular automata but where the same manufacturers are unable to construct reversible computers for other purposes. On the other hand, if our RCO-POW used instead a random reversible circuit that change completely every new block, then manufacturers will initially find it much more difficult to construct a chip that computes a new circuit every block than if the problem were based on a reversible cellular automaton. However, since our

RCO-POW problem scheme uses five different problems of different manufacturing difficulty levels, a reversible hardware manufacturer could first construct a device for solving the Problem 1 (which should be easier to construct). After manufacturers construct reversible devices for solving Problem 1, they will have the expertise needed to construct torus shaped reversible devices for solving Problem 2. The manufacturers will then gradually innovate their reversible computing technology until they solve Problems 1-5. After they construct devices for solving Problems 1-5, these manufacturers will be able to construct reversible computers for purposes other than cryptocurrency mining.

Problems 1-3 have a higher reversibility ratio than problems 4-5 which means that reversible computers will spend more logic gates computing the reversible functions F_1, F_2, F_3 rather than performing the irreversible operations of verifying the solution and setting up the initial input for the next attempt at a solution. Therefore, the high reversibility ratio for F_1, F_2, F_3 will further increase the ease of solving Problems 1-3.

1.1. **Some Nebula specifications:** The specifications for Nebula are tentative and will only be set in stone when Nebula is launched.

Halving time: 8 years.

Average block time: A new block for Nebula will be formed on average every 2 minutes. The means the difficulty for each problem in R5 will be adjusted so that Problem i is solved on average every 10 minutes.

What will the coins be called? Each coin shall be called a CIRC which stands for Certificate of Innovation in Reversible Computation. The symbol for Nebula shall be CIRC.

Innovation and market bonus: The innovation and market bonus will be an increase in the block reward that is supposed to take effect when Nebula obtains a high market cap and when reversible computing devices begin to be developed in order to solve R5. If the block reward without the IMB is K CIRCs, then the block reward with the IMB will be $K \cdot (1 + \frac{\max(j,15)}{15})$ CIRCs where j is the least nonnegative integer where the average hash rate per block is less than 2^{75+j}.

1.2. **Considerations.** In this section, we shall address some of the concerns about R5 and Nebula.

Attack immunity: Since R5 uses five different algorithms, it will be difficult to launch an attack against Nebula even if one simply finds an insecurity in one of the problems in R5. After all, one will not be able to formulate a 51 percent attack against Nebula simply by obtaining a secret algorithm for solving Problem 1 quickly. The Nebula community will also reject all forks where a majority of the highest blocks are blocks where Problem 1 is solved instead of Problems 2-5. More drastically, the Nebula miners may also choose to temporarily reject all blocks that solve Problem 1 until the security weakness is rectified. In the scenario where an entity obtains a secret algorithm for solving Problem 1 and compromises the security of R5, we would have to fork Nebula in order to increase the number of rounds in Problem 1 so that it remains secure. Of course, such a fork caused from a security weakness will result in a dip in the market cap for Nebula and other deleterous effects even if Nebula is not directly harmed by an attack, so R5 is designed so that no security weakness is conceivable in the foreseeable future.

Why the long 8 year halving period? R5 as implemented in Nebula shall have an 8 year halving period rather than the 4 year halving period for Bitcoin. However, the history of Bitcoin teaches us that Bitcoin's halving period is just too short since it takes more than 4 years for the cryptocurrency market cap to become half-way saturated. This four year halving period has resulted in most of the bitcoins being owned by very few individuals. With an 8 year halving period, Nebula will be distributed more evenly so that the early adopters are rewarded for making Nebula more well-known while Nebula retains a fair distribution of wealth.

A long halving period will also improve the reversible computer friendliness of Nebula. It is believed that reversible computers must come into dominance in the 2030's when the efficiency of conventional computers is restricted by Landauer's limit. Therefore, if Nebula was launched in 2018 and had a 4 year halving period, then the initial distribution of Nebula through mining will be mostly completed by the time that the reversible computer industry is starting to arise. However, with an 8 year halving period, Nebula mining will still be a strong market by the time reversible computers come into dominance.

Specialized hardware resistance and manufacturing centralization: Since until reversible devices are in widespread general purpose use, there cannot be an RCO-POW problem which is resistant to specialized hardware. However, Problem 4 and Problem 5 in Nebula can be considered to be somewhat ASIC-resistant since the circuit that solves Problem 4 changes completely annually and the circuit that solves Problem 5 changes slightly after every block. R5 is also partially ASIC resistant in the sense that it will take a greater amount of time to create ASICs for five problems than it would to create ASICs for only one problem.

256 bits erased to initialize each hash:

While the POW scheme R5 is far more reversible than all the other POW problems used in other cryptocurrencies, the POW scheme R5 requires irreversibility or uncomputation in order to reset the data after each hash. Suppose that a miner has just computed $f_i(\mathbf{k}\#\mathbf{x})$ and is now going to the next hash attempt. Suppose also that $f_i(\mathbf{k}\#\mathbf{x}) = \mathbf{k}'\#\mathbf{y}$. Then the miner needs to overwrite $\mathbf{k}'$ with $\mathbf{k}$ which requires the miner to erase 256 bits of information (unless the miner uncomputes the function f_i after every solution attempt). However, the miner does not need to overwrite $\mathbf{y}$ since the miner can compute $f_i(\mathbf{k}\#\mathbf{y})$ in her next attempt at solving the POW problem.

We say that a RCO-POW problem P is complete if there is an efficient reversible algorithm A for solving P where not even the output data is overwritten by input data after each hash and where the best algorithm for solving problem P differs from A by very few gates.

We recognize that it may be possible to construct a complete RCO-POW problem, but we have not been able to construct any complete RCO-POW problems and the construction of a complete RCO-POW problem may be a difficult cryptography research problem. Fortunately, there is a sense in which incomplete RCO-POW problems incentivize the construction of the reversible computer just as well as complete RCO-POW problems. A complete RCO-POW problem simply incentivizes the construction of a reversible device, but an incomplete RCO-POW problem incentivizes the construction of a reversible device which contains an irreversible component which resets data or is at least connected to a device capable of resetting data.

Other cryptocurrencies: The proof-of-work problem R5 or a modification thereof may be used as the POW for other cryptocurrencies as well in the future as cryptocurrencies switch towards useful POWs. If a cryptocurrency is to use an RCO-POW problem for its POW, then we suggest to use only a slight modification of R5 or complete RCO-POW problems so that chip manufacturers can focus on constructing chips to solve these specific kinds of problems instead of constructing a broad class of chips.

Why not 1D or 3D reversible cellular automata or more problems?

The problem with 1D cellular automata is that it typically takes many more rounds for a 1D cellular automata to achieve security than for other RCO-POW problems. Therefore, validating a 1D cellular automata based RCO-POW problem will take much more effort than validating Problems 1-5, and a 1D cellular automata based RCO-POW problem will likely require the circuits to either be fairly error-free or to have very good error correction techniques in place. On the other hand, a 3D reversible cellular automaton will need more layers per round, be more complex, and be harder to visualize than a 2D reversible cellular automaton.

Keep in mind that in the future various cryptocurrencies may use different RCO-POW problems, so the number of RCO-POW problems a chip manufacturer must select from will increase as time goes by. If there are fewer RCO-POW problems, then chip manufacturers can focus all of their efforts on solving these few problems rather than many problems. We shall therefore refrain from constructing a RCO-POW based on a cellular automaton on a cube, Mobius strip, Klein bottle, or $S^1 \times S^1 \times S^1$.

The innovation and market bonus:

The idea behind the IMB is that the IMB will encourage the development of reversible computation as the efficiency of reversible computation approaches Landauer's limit and as the market cap for Nebula becomes extremely large. The mining reward for Nebula will begin to increase when the hash rate per block reaches 2^{75} and the mining reward will become twice the original reward when the hash rate per block reaches 2^{90}.

Take note that as of October 23, 2017 the mining reward for each 10 minute block in Bitcoin is 75000 USD which comes out to 15000 USD every 2 minutes. At 5 cents per KWH, if the efficiency per gate in R5 is Landauer's limit, then at temperature 300 K for Problem 2, if 15000 USD are spent per 2 minute block, then the hash rate will be $7.36954 \cdot 10^{27} = 2^{92.5}$. Therefore, the IMB will take effect as the mining reward for Nebula approaches that of Bitcoin in October 23, 2017 and as the efficiency of devices that solve R5 approach Landauer's limit.

What is optimization freeness? A POW A is said to be optimization free if there is no possibility for a faster algorithm for solving A than the standard algorithm. For Problem i in R5, the standard algorithm is to first compute a one hash $\mathbf{k}$ and then try many possible 68 bit strings $\mathbf{x}$. For the standard algorithm for solving Problem i, one needs to compute the hash $\mathbf{k}$ as seldomly as possible since recomputing the hash $\mathbf{k}$ takes work. Therefore, in order for Problem i to remain optimization free, there should be no possible way to search for a good hash $\mathbf{k}$ for which Problem i is easier to solve. Optimization freeness is the main security requirement for POW problems.

1.3. **Using cryptocurrencies to ensure security.** Cryptocurrencies themselves may be used to ensure that their own POW problems are secure using the following

technique which we shall call the *internal testing technique*. The idea behind the internal testing technique is that miners occasionally will be given a block reward for breaking weakened versions of the POW problem and that the security level of the POW problem will automatically increase as weakened POW problems are broken.

Let P_n be the n-round version of a certain POW problem whose security level is still being investigated. Suppose furthermore that the POW problem P_{SAFE} is secure for some sufficiently large natural number SAFE (most symmetric cryptosystems can be made secure simply by performing a sufficient number of rounds). Suppose that the objective of the POW problem P_n is to find some string $\mathbf{x}$ such that $f_n(\mathbf{k}\#\mathbf{x}) < d^{-1}$ where d is the difficulty, $\mathbf{k}$ is the hash of the header and $\mathbf{x}$ is a string.

The following scenario outlines a possible instance of how the internal testing technique can be implemented. When the cryptocurrency which seeks to implement the internal testing technique is launched, one initially chooses n to be the least natural number so that the POW problem P_n has not been broken yet. The cryptocurrency will employ the following two versions of the POW problem.

Ordinary POW: Find some string $\mathbf{x}$ and a hash of the block header $\mathbf{k}$ so that $f_{3n}(\mathbf{k}\#\mathbf{x}) < d^{-1}$ where d is the difficulty.

Weakened security POW: Find some string $\mathbf{x}$ and an admissible hash $\mathbf{k}$ so that $f_n(\mathbf{k}\#\mathbf{x}) < \frac{d^{-1}}{100000}$.

For each block, miners are allowed to solve either the ordinary POW problem or the weakened security POW problem, and the difficulty is adjusted so that a new block is created every 2 minutes. For every 24 hour period, if the occurrence of a solution to the weakened security POW is found at least 1/100 as often as a solution to the ordinary POW, then increment n by one, otherwise decrement n by one.

One advantage of the internal testing technique is that the weakened security POW is actually a useful POW problem with scientific importance.

We ultimately have decided not to employ the internal testing technique since we have already seen that the security requirements for R5 are already quite low.

2. Analysis of the problems

The problems in R5 are classified into cellular automata-like problems (Problems 1-3) and random circuit problems (Problems 4-5). The security of Problems 1-3 is verified mostly experimentally while the security of Problems 4-5 is verified using Markov chains and their generalizations.

2.1. General facts about R5.

The reversibility ratio of the a POW problem is the number of gates used in the circuit computing the function F_i to the number of bits erased after every hash attempt (in the case of R5, after every hash attempt, 256 bits are erased).

The support of a logic gate $f : \{0,1\}^I \to \{0,1\}^I$ is the smallest set $\text{supp}(f) = R$ such that there is some function $\hat{f} : \{0,1\}^R \to \{0,1\}^R$ such that if $f(x_i)_{i\in I} = (y_i)_{i\in I}$, then $\hat{f}(x_r)_{r\in R} = (y_r)_{r\in R}$ and $x_s = y_s$ for $s \in I \setminus R$. We shall say that a sequence of gates $f_1, \ldots, f_n : \{0,1\}^I \to \{0,1\}^I$ is a layer if $\text{supp}(f_i) \cap \text{supp}(f_j) = \emptyset$ whenever $i, j \in \{1, \ldots, n\}$.

In R5, all logic gates under consideration are Toffoi*, Fredkin and CNOT gates.

Function	σ	τ	μ
Total gates	162	81	256
Layers	2	1	4

Problem number	1	2	3	4	5
Nonlinear gate type	Fredkin	Toffoli*	Both	Fredkin	Toffoli*
Total gates:F_i	46,899	52,339	46,899	39,379	25,075
Total gates:f_i	46,400	51,840	46,400	38,880	24,576
CNOT gates F_i	37,619	41,971	37,619	19,939	499
CNOT gates f_i	37,120	41,472	37,120	19,440	0
Toffoli* gates	0	10,368	4,640	0	24,576
Fredkin gates	9,280	0	4,640	19,440	0
Number of rounds	64	64	64	180	24,576
Gates per round	725	810	725	216	1
Gates per layer in f_i	64-81	81	64-81	108	Varies
Layers per round	10	10	10	2	Varies
Layers in f_i	640	640	640	360	Varies
Layers in F_i	647	647	647	367	Varies
Toffoli* per round	0	162	∼72.5	0	1
Fredkin per round	145	0	∼72.5	108	0
CNOT per round	580	648	580	108	0
Reversibility ratio	183.199	204.449	183.199	153.82	97.95

2.2. **Overview of τ, μ, σ.** The functions σ, τ have been implemented into R5 in order to help achieve optimization freeness and to boost the security of R5 without adding too many gates to R5. Without the functions σ, τ, a miner could solve the problems in R5 using circuits smaller than the standard circuits for solving R5 since some of the gates in the standard circuit for solving R5 will only affect insignificant final bits in the output. However, it is unlikely for such a strategy to work with the function σ, τ in place since the control bits for the CNOT gates in the functions σ, τ are on the insignificant output bits while the output bits for the CNOT gates in σ, τ are the first 81 most significant output bits. The functions σ, τ also improve

the security of R5 since they increase the unpredictability of the first 81 output bits and to a lesser extent the first 162 output bits.

The function μ has been implemented in order to evenly amplify the rate at which differences in the string $\mathbf{x}$ will result in differences in the output $f_i(\mathbf{k}\#\mathbf{x})$. For each bit that one changes in the input $\mathbf{x}$ between 4 and 5 bits are changed in the string $\mu(\mathbf{k}\#\mathbf{x})$. Furthermore, if one bit is changed within the string $\mathbf{x}$, then the 4 or 5 bits in the array $\Sigma \circ \mu(\mathbf{k}\#\mathbf{x})$ which are changed will be spaced fairly evenly throughout $\{0,\dots,17\}\times\{0,\dots,17\}$, and by the following proposition, at least one of these 4 to 5 changed bits in $\Sigma \circ \mu(\mathbf{k}\#\mathbf{x})$ will be fairly distant from the boundary of $\{0,\dots,17\}\times\{0,\dots,17\}$.

Proposition 2.1. *Let $\mathbf{k}$ be a 256 bit string and let $\mathbf{x},\mathbf{y}$ be distinct 68 bit strings. Let $(x_{i,j})_{0\leq i<17,0\leq j<17} = \Sigma\circ\mu(\mathbf{k}\#\mathbf{x})$ and let $(y_{i,j})_{0\leq i<17,0\leq j<17} = \Sigma\circ\mu(\mathbf{k}\#\mathbf{y})$ Then there are some $i,j\in\{3,\dots,14\}$ where $x_{i,j}\neq y_{i,j}$.*

2.3. **Problems 1-3: Cellular automata like problems.** Suppose that

$$C:\{0,1\}^{2\times 2}\to\{0,1\}^{2\times 2}$$

is a permutation. Then define mappings

$$E_C, O_C : \{0,1\}^{18\times 18}\to\{0,1\}^{18\times 18}$$

from the permutation C by letting $E_C(x_{i,j})_{i,j} = (y_{i,j})_{i,j}$ precisely when

$$C((x_{2\cdot r+i \mod 18, 2\cdot s+j \mod 18})_{i,j\in\{0,1\}})$$

$$= (y_{2\cdot r+i \mod 18, 2\cdot s+j \mod 18})_{i,j\in\{0,1\}}$$

for all integers r,s and $O_C(x_{i,j})_{i,j} = (z_{i,j})_{i,j}$ precisely when

$$C((x_{1+2\cdot r+i \mod 18, 1+2\cdot s+j \mod 18})_{i,j\in\{0,1\}})$$

$$= (z_{1+2\cdot r+i \mod 18, 1+2\cdot s+j \mod 18})_{i,j\in\{0,1\}}$$

for all integers r,s.

Let $O'_C:\{0,1\}^{18\times 18}\to\{0,1\}^{18\times 18}$ be the mapping where $O_C(x_{i,j})_{i,j} = (z_{i,j})_{i,j}$ precisely when

$$C((x_{1+2\cdot r+i,1+2\cdot s+j})_{i,j\in\{0,1\}}) = (z_{1+2\cdot r+i,1+2\cdot s+j})_{i,j\in\{0,1\}}$$

for $r\in\{0,\dots,7\}$ and $z_{i,j} = x_{i,j}$ whenever $\{i,j\}\cap\{0,17\}\neq\emptyset$.

Then the functions $E_C\circ O_C, E_C\circ O'_C : \{0,1\}^{18\times 18}\to\{0,1\}^{18\times 18}$ are reversible cellular automata commonly known as a block cellular automata. The permutation C shall be called the fundamental permutation for the block cellular automata $E_C\circ O_C, E_C\circ O'_C$.

Recall that Problem 1 requires one to find exceptionally low values $\tau\circ\sigma\circ(E_C\circ O'_C)^{64}\circ\mu(\mathbf{k}\#\mathbf{x})$ and Problem 2 requires one to find exceptionally low values $\tau\circ\sigma\circ(E_D\circ O_D)^{64}\circ\mu(\mathbf{k}\#\mathbf{x})$ where $\mathbf{k}$ is a 256 bit hash and $\mathbf{x}$ is a 68 bit string. The fundamental permutations C,D should satisfy the following characteristics in order to ensure the security of the Problem 1 and Problem 2.

Visual tests-One can view these block cellular automata at

$$https://dmishin.github.io/js-revca/index.html$$

in order to detect possible security deficiencies in these cellular automata. The cellular automata generated by the fundamental permutations C,D should be Class 3

cellular automata according to Wolfram's 4 classes of cellular automata. Furthermore, changes in the input of the cellular automata should spread across the grid as rapidly as possible. The permutations C, D satisfy this property.

Non-conservativity-A permutation C is said to be conservative if whenever $C((x_{i,j})_{i,j} = (y_{i,j})_{i,j}$, we have

$$x_{0,0} + x_{0,1} + x_{1,0} + x_{1,1} = y_{0,0} + y_{0,1} + y_{1,0} + y_{1,1}.$$

The permutations C, D should be non-conservative since conservativity reduces the security of Problems $1, 2$. Fortunately, most permutations including C, D are non-conservative.

Few gates-The fundamental permutations must consist of as few logic gates as possible. Furthermore, most of these gates should be CNOT gates. A fundamental permutation that requires many gates will undoubtably make it more difficult to construct a reversible device for solving Problems 1-2. It will be more difficult to construct Toffoli* and Fredkin gates than it will to construct NOT and CNOT gates, so we shall use Toffoli* and Fredkin gates sparingly in Problems 1-2.

Variety-The fundamental permutations C, D should be sufficiently different in order to incentivize a variety of reversible devices. The circuit computing fundamental permutation C has one Fredkin gate while the circuit computing D has one Toffoli gate.

Non-linearity-The permutations C, D must be non-linear as a permutation of the vector space $\mathbb{F}_2^4$. Non-linearity is required in order to thwart any possible linear algebraic attacks against Problems 1-2. Therefore, the circuits computing the permutations C, D must consist of at least one non-linear gate such as Toffoli* gates or Fredkin gates.

Opposite corner property-The permutations C, D must satisfy the opposite corner property. A permutation C is said to have the opposite corner property if whenever $r, s \in \{0, 1\}$ and $x_{i,j} = y_{i,j}$ for $(i, j) \neq (r, s)$ and $x_{r,s} \neq y_{r,s}$ and $C(x_{i,j})_{i,j} = (x'_{i,j})_{i,j}$, $C(y_{i,j})_{i,j} = (y'_{i,j})_{i,j}$, then $x'_{1-r,1-s} \neq y'_{1-r,1-s}$. The opposite corner property is required in order for ensure that a change in the input to the corresponding cellular automata propogates as quickly as possible throughout the grid. Without the opposite corner property, it will take many more rounds to ensure that Problem 1 and Problem 2 are secure.

High switching coefficient-The permutations C, D should have a high switching coefficient. The switching coefficient of the permutation C is the probability that $x'_{r,s} \neq y'_{r,s}$ where r, s are randomly selected from $\{0, 1\}$,

$$C(x_{i,j})_{i,j} = (x'_{i,j})_{i,j}, C(y_{i,j})_{i,j} = (y'_{i,j})_{i,j},$$

and where each $x_{i,j}, y_{i,j}$ is randomly selected subject to the condition that

$$|\{(i, j)|x_{i,j} \neq y_{i,j}\}| = 1.$$

A high switching coefficient is required to ensure that changes to the input to the corresponding cellular automata quickly propogate throughout the grid.

Corner entrance and corner escape property–The permutation C should satisfy the corner entrance and corner escape property (the permutation D is not required to satisfy the corner entrance property nor the corner entrance property since the grid for the cellular automaton for Problem 2 is shaped as a torus).

Let $T_{r,s}$ be the permutation of $\{0, 1\}^{2\times 2}$ where $T_{r,s}(x_{i,j})_{i,j} = (y_{i,j})_{i,j}$ precisely when $x_{i,j} = y_{i,j}$ for $(i, j) \neq (r, s)$ and $y_{r,s} = 1 - x_{r,s}$.

A permutation C satisfies the corner escape property if whenever $r,s \in \{0,1\}$, if $(x_{i,j})_{i,j} \neq (y_{i,j})_{i,j}$ there are $J_0,\dots,J_{N-1}$ such that for $0 \leq n < N$ either $J_n = T_{r,s}$ or

$$J_n : \{0,1\}^{2\times 2} \to \{0,1\}^{2\times 2}$$

is the identity function and if

$$(J_{N-1} \circ C) \circ \cdots \circ (J_0 \circ C)(x_{i,j}) = (x'_{i,j})_{i,j}$$

and

$$(J_{N-1} \circ C) \circ \cdots \circ (J_0 \circ C)(y_{i,j}) = (y'_{i,j})_{i,j},$$

then $x'_{r,s} = y'_{r,s}$. The permutation C for Problem 1 satisfies the corner escape property.

A permutation C satisfies the level N corner entrance property for (r,s) if for all $\mathbf{x},\mathbf{y}$ there are J_n for $n < N$ where $J_n = T_{r,s}$ or J_n is the identity permutation and where $(C \circ J_{N-1}) \circ \cdots \circ (C \circ J_0)(\mathbf{x}) = \mathbf{y}$. A permutation C satisfies the corner entrance property for (r,s) if it satisfies the level N corner entrance property for (r,s) for some N. The permutation C satisfies the corner entrance property for corners $(1,0),(0,0)$, but C does not satisfy the corner entrance property for corners (0,1) and (1,1). While the corner entrance property and corner escape property are both desirable features for the permutation C, we were willing to compromise the corner entrance property for (0,1) and (1,1) in order to obtain a permutation that satisfies other desirable properties.

The computer test on Problems 1-3. We shall now give the results of the main computer test for the security of Problems 1-3. The main computer test for the security of Problems 1-3 is to evaluate the probabilistic functions Z_1, Z_2, Z_3 each 100,000,000 times. Informally, the function Z_i counts the number of generations it takes for a one bit change in the input to a reversible cellular automaton corresponding to Problem i to propagate throughout the entire 18×18 grid.

The function Z_w for $w \in \{1,2,3\}$: Let $(r,s) \in \{2,\dots,15\} \times \{2,\dots,15\}$ be randomly selected. Let $(x_{i,j})_{i,j\in\{0,\dots,17\}}, (y_{i,j})_{i,j\in\{0,\dots,17\}} \in \{0,1\}^{18\times 18}$ be randomly selected subject to the conditions that $x_{i,j} = y_{i,j}$ for $(i,j) \neq (r,s)$ and $x_{r,s} \neq y_{r,s}$. Let

$$x^0_{i,j} = x_{i,j}, (x^n_{i,j}{}')_{i,j} = O'_C(x^n_{i,j})_{i,j}), x^{n+1}_{i,j} = E_C((x^n_{i,j})_{i,j}),$$
$$y^0_{i,j} = y_{i,j}, (y^n_{i,j}{}')_{i,j} = O'_C(y^n_{i,j})_{i,j}), y^{n+1}_{i,j} = E_C((y^n_{i,j})_{i,j}),$$
$$xx^0_{i,j} = x_{i,j}, (xx^n_{i,j}{}')_{i,j} = O_D(xx^n_{i,j})_{i,j}), xx^{n+1}_{i,j} = E_D((xx^n_{i,j})_{i,j}),$$
$$yy^0_{i,j} = y_{i,j}, (yy^n_{i,j}{}')_{i,j} = O_D(yy^n_{i,j})_{i,j}), yy^{n+1}_{i,j} = E_D((yy^n_{i,j})_{i,j}),$$
$$xxx^0_{i,j} = x_{i,j}, (xxx^n_{i,j}{}')_{i,j} = LO_n(xxx^n_{i,j})_{i,j}), xxx^{n+1}_{i,j} = LE_n((xxx^n_{i,j})_{i,j}),$$
$$yyy^0_{i,j} = y_{i,j}, (yyy^n_{i,j}{}')_{i,j} = LO_n(yyy^n_{i,j})_{i,j}), yyy^{n+1}_{i,j} = LE_n((yyy^n_{i,j})_{i,j}).$$

Let Z_1 return the least natural number N such that for all but at most two $(r,s) \in \{0,\dots,7\} \times \{0,\dots,7\}$ there is some $n \in \{0,\dots,N-1\}$ where

$$(x^n_{1+2r+i,1+2s+j}{}')_{i,j\in\{0,1\}} \neq (y^n_{1+2r+i,1+2s+j}{}')_{i,j\in\{0,1\}}$$

or

$$(x^{n+1}_{1+2r+i,1+2s+j})_{i,j\in\{0,1\}} \neq (y^{n+1}_{1+2r+i,1+2s+j})_{i,j\in\{0,1\}}.$$

Let Z_2 return the least natural number N such that for all but at most two $(r,s) \in \{0,\dots,8\} \times \{0,\dots,8\}$ there is some $n \in \{0,\dots,N-1\}$ where

$$(xx^n_{2r+i,2s+j}{}')_{i,j\in\{0,1\}} \neq (yy^n_{2r+i,2s+j}{}')_{i,j\in\{0,1\}}$$

or

$$(xx^{n+1}_{2r+i,2s+j})_{i,j\in\{0,1\}} \neq (yy^{n+1}_{2r+i,2s+j})_{i,j\in\{0,1\}}.$$

Let Z_3 return the least natural number N such that for all but at most two $(r,s) \in \{0,\ldots,7\} \times \{0,\ldots,7\}$ there is some $n \in \{0,\ldots,N-1\}$ where

$$(xxx^{n}_{1+2r+i,1+2s+j}{}')_{i,j\in\{0,1\}} \neq (yyy^{n}_{1+2r+i,1+2s+j}{}')_{i,j\in\{0,1\}}$$

or

$$(xxx^{n+1}_{1+2r+i,1+2s+j})_{i,j\in\{0,1\}} \neq (yyy^{n+1}_{1+2r+i,1+2s+j})_{i,j\in\{0,1\}}.$$

We have computed the probabilistic functions Z_1, Z_2, Z_3 100,000,000 times. The following table summarizes the outputs of the functions Z_1, Z_2, Z_3.

Problem	$w=1$	$w=2$	$w=3$
Max Z_w	15	8	13
$Z_w=4$	1940106	14264905	2346497
$Z_w=5$	8263666	78477920	15878786
$Z_w=6$	28915804	7230493	35013876
$Z_w=7$	30568467	26614	32794017
$Z_w=8$	22012425	68	12809757
$Z_w=9$	7095395		1041009
$Z_w=10$	1086101		112395
$Z_w=11$	107142		3582
$Z_w=12$	9729		78
$Z_w=13$	1047		3
$Z_w=14$	112		
$Z_w=15$	6		

Since a one bit difference of $(x_{i,j})_{i,j\in\{0,\ldots,17\}}$ located in $\{2,\ldots,15\}^2$ results in a completely different output after 16 rounds and since Problems 1-3 use 64 rounds, we conclude that Problems 1-3 are optimization free and secure for public use in cryptocurrencies.

2.4. **Problems 4-5: Random circuit problems.** Recall that the function F_5 is a function consisting of 24,576 random Toffoli* gates on 324 bits. The Toffoli* gate is universal for reversible computation and hence also reversible for classical computation. Due to the simplicity of F_5, we shall be able to analyze the security of F_5 mathematically using Markov chains. Similarly, we shall also be able to analyze the security of F_4 using Markov chains and similar structures.

Markov chain calculations: Suppose that $x \in \{1,\ldots,323\}$. Let P_x, P_x^-, P_x^+ denote the following probabilities. Suppose that

$$(r_i)_{i\in\{0,\ldots,323\}}, (s_i)_{i\in\{0,\ldots,323\}} \in \{0,1\}^{324}$$

are selected at random subject to the condition that $\{i \in \{0,\ldots,323\} | r_i \neq s_i\}$. Let $T : \{0,1\}^{324} \to \{0,1\}^{324}$ be a randomly selected Toffoli* gate. Suppose now that

$$(r'_i)_{i\in\{0,\ldots,323\}} = T((r_i)_{i\in\{0,\ldots,323\}}),$$

$$(s'_i)_{i\in\{0,\ldots,323\}} = T((s_i)_{i\in\{0,\ldots,323\}}).$$

Then define P_x to be the probability that $\{i \in \{0,\ldots,323\} | r'_i \neq s'_i\} = x$, define P_x^- to be the probability that $\{i \in \{0,\ldots,323\} | r'_i \neq s'_i\} = x-1$, and define P_x^+ to be the probability that $\{i \in \{0,\ldots,323\} | r'_i \neq s'_i\} = x+1$.

Let V be the real-valued vector space generated by the basis $e_1,\dots,e_{162}$. Let $L:V\to V$ be the linear transformation where $L(e_1)=P_1e_1+P_1^+e_2$ and where $L(e_x)=P_x^-\cdot e_{x-1}+P_x\cdot e_x+P_x^+\cdot e_{x+1}$ for $1\leq x<162$, and let $L(e_{162})=e_{162}$. Then L is a stochastic linear transformation with unique stationary distribution e_{162}. The motivation behind the linear mapping L is that if $L^n(e_1)=a_1e_1+\cdots+a_{162}e_{162}$, then a_{162} is the probability that if $(x_i)_{i\in\{0,\dots,323\}},(y_i)_{i\in\{0,\dots,323\}}\in\{0,1\}^{324}$ are random bitstrings that differ by one bit and $T_0,\dots,T_{n-1}:\{0,1\}^{324}\to\{0,1\}^{324}$ are random Toffoli* gates, then there is some $m<n$ where $T_m\circ\cdots\circ T_0(x_i)_{i\in\{0,\dots,323\}}$ and $T_m\circ\cdots\circ T_0(y_i)_{i\in\{0,\dots,323\}}$ differ in at least 162 different places. More informally, a_{162} is the probability that if $(x_i)_{i\in\{0,\dots,323\}},(y_i)_{i\in\{0,\dots,323\}}\in\{0,1\}^{324}$ are random bitstrings that differ by one bit, then

$$T_n\circ\cdots\circ T_0(x_i)_{i\in\{0,\dots,323\}}$$

and

$$T_n\circ\cdots\circ T_0(y_i)_{i\in\{0,\dots,323\}}$$

have no correlation with each other.

Let $i_x=\frac{(324-x)(323-x)}{324\cdot 323},j_x=\frac{2\cdot(324-x)\cdot x}{324\cdot 323},k_x=\frac{x\cdot(x-1)}{324\cdot 323}$. Then
$P_x^-=\frac{k_x}{2}\cdot\frac{x-2}{322}+\frac{j_x}{2}\cdot\frac{x-1}{322}$,
$P_x=i_x+\frac{j_x}{2}+\frac{k_x}{2}$,
$P_x^+=\frac{k_x}{2}\cdot\frac{324-x}{322}+\frac{j_x}{2}\cdot\frac{323-x}{322}$.

Our computer tests indicate that 24260 is the least natural number such that $\|L(e_1)-e_{162}\|<2^{-67}$. We shall call 24260 the 2^{-68}-reversal gate count of F_5. Since Problem 5 uses 24,576 Toffoli gates, we conclude that the function F_5 is a good randomizing function in the sense that a one bit change of $(x_i)_{i\in\{0,\dots,323\}}$ results in a completely different output $F_5((x_i)_{i\in\{0,\dots,323\}})$.

Let $E_1(x,y,z)=(x\oplus y,y,z)$ and $E_2(0,y,z)=(0,y,z),E_2(1,y,z)=(1,z,y)$. Let $E=E_1\circ E_2$. The gate E shall be called an enhanced gate. Fredkin gates are universal for reversible computation, but Fredkin gates alone are not sufficient to achieve cryptographic security since Fredkin gates preserve the number of 1 bits and the number of 0 bits. Recall that F_4 consists of $19,440$ enhanced gates. Using the same Markov chain calculation as we have performed for F_5, we have calculated the 2^{-68}-reversal gate count of F_4 to be 19516. Since $\lceil 19516/108\rceil=181$, we conclude that 180 rounds for F_4 should make F_4 cryptographically secure. Since the gates in F_4 are neatly partitioned into layers, one can improve the Markov chain calculation and conclude that a one bit change of the input for F_4 will result in a ≥ 162 bit change in the intermediate output before all 163 rounds are calculated.

References

E-mail address: jvanname@mail.usf.edu